

Дано два файла – MainProgramm и лаунчер

Пробуем запустить MainProgramm

```
Enter key: 121234
INVALID
Access denied. Launcher did not authorize (exit code = 1).
|
```

Видимо, она что-то делает с лаунчером

Пытаемся запустить лаунчер

```
Enter key: 23123|
```

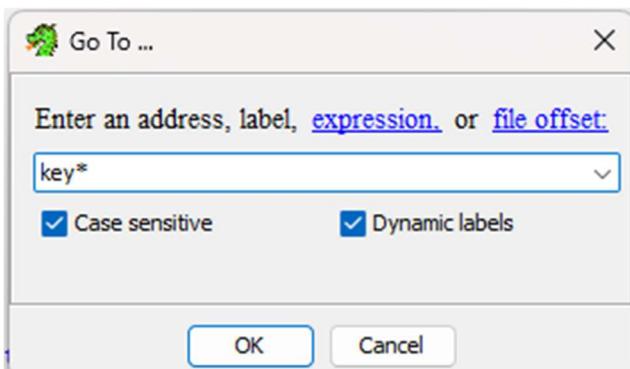
Требует запустить ключ полностью аналогичным выводом и после ввода закрывается

Идём в гидру смотреть лаунчер

Сразу найденная точка входа ничего не дает

```
1
2 /* Library Function - Single Match
3     int __cdecl __scrt_common_main(void)
4
5     Libraries: Visual Studio 2017 Debug, Visual Studio 2019 Debug */
6
7 int __cdecl __scrt_common_main(void)
8
9 {
10     int iVar1;
11
12     __security_init_cookie();
13     iVar1 = __scrt_common_main_seh();
14     return iVar1;
15 }
```

Вспоминаем вывод в консоль, ищем



Находим дизассемблированный участок, там присутствует XREF[1]: FUN...

<pre>s_Enter_key:_1400227c0 1400227c0 45 6e 74      ds      "Enter key: "           65 72 20           6b 65 79 ...</pre>	<pre>XREF[1]:     FUN_140017f10:140017f58(*)</pre>
<pre>1400227cc 00            ??      00h 1400227cd 00            ??      00h 1400227ce 00            ??      00h 1400227cf 00            ??      00h</pre>	

Идем туда, смотрим дизассемблированный код. Сразу можно заметить ветвление на OK и INVALID-статусы проверки

```
if (cVar2 == '\0') {
    thunk_FUN_140012030((longlong *)cout_exref,"INVALID\n");
    local_64 = 1;
    thunk_FUN_140014f60(local_200);
    thunk_FUN_140014f60(local_240);
}
else {
    thunk_FUN_140012030((longlong *)cout_exref,"OK\n");
    local_84 = 0;
    thunk_FUN_140014f60(local_200);
    thunk_FUN_140014f60(local_240);
}
```

Планируем пропатчить лаунчер, чтобы проверка всегда была успешной

Ищем адрес инструкции

<pre>140018017 0f b6 c0      MOVZX   EAX,AL 14001801a 85 c0      TEST    EAX,EAX 14001801c 74 3e      JZ     LAB_14001805c 14001801e 48 8d 15      LEA    RDX,[DAT_1400227d0]           ab a7 00 00 140018025 48 9b 0d      MOV    RCX,qword ptr [-MSVCP140D.DLL::std::cout]           dc 31 01 00 14001802c e8 8d 90      CALL   thunk_FUN_140012030           ff ff 140018031 90            NOP 140018032 c7 85 c4      MOV    dword ptr [RBP + local_84],0x0           01 00 00</pre>	<pre>i 42 43 44 45 46 47 48 49 50 51 52 53</pre> <pre>cVar2 = thunk_FUN_1400124d0((longlong)local_240,(longlong)local_20 if (cVar2 == '\0') {     thunk_FUN_140012030((longlong *)cout_exref,"INVALID\n");     local_64 = 1;     thunk_FUN_140014f60(local_200);     thunk_FUN_140014f60(local_240); } else {     thunk_FUN_140012030((longlong *)cout_exref,"OK\n");     local_84 = 0;     thunk_FUN_140014f60(local_200);     thunk_FUN_140014f60(local_240);</pre>
--	---

Имеет опкод 74 3e, находится по адресу 1801c, так как 1400 – виртуальная часть, в системе база модуля будет загружена по другому адресу, но в этом модуле нужная инструкция будет находиться именно по 1801c

Идём в x64dbg. Открываем лаунчер, жмем Alt+e, ищем модуль launcher.exe имеет базу 00007FF6F6620000. Прибавляем к нему 1801c, получим 00007FF6F663801C

Идем по этому адресу, видим следующее

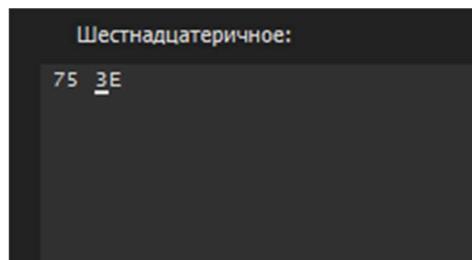
```
    74 3E      je launcher._FFF6F663805C
48:8D15 ABA70000  lea rdx,qword ptr ds:[<"OK\n"...
48:8B0D DC310100  mov rcx,qword ptr ds:[<class std::basic_ostream<char, struct std::
call launcher._FFF6F66310BE
90          nop
C785 C4010000 0000  mov dword ptr ss:[rbp+1C4],0
48:8D4D 48      lea rcx,qword ptr ss:[rbp+48]
E8 FB90FFFF  call launcher._FFF6F6631140
90          nop
48:8D4D 08      lea rcx,qword ptr ss:[rbp+8]
E8 F190FFFF  call launcher._FFF6F6631140
8885 C4010000  mov eax,dword ptr ss:[rbp+1C4]
E9 9B000000  jmp launcher._FFF6F66380F5
E8 39          jmp launcher._FFF6F6638095
48:8D15 75A70000  lea rdx,qword ptr ds:[<"INVALID\n"...

```

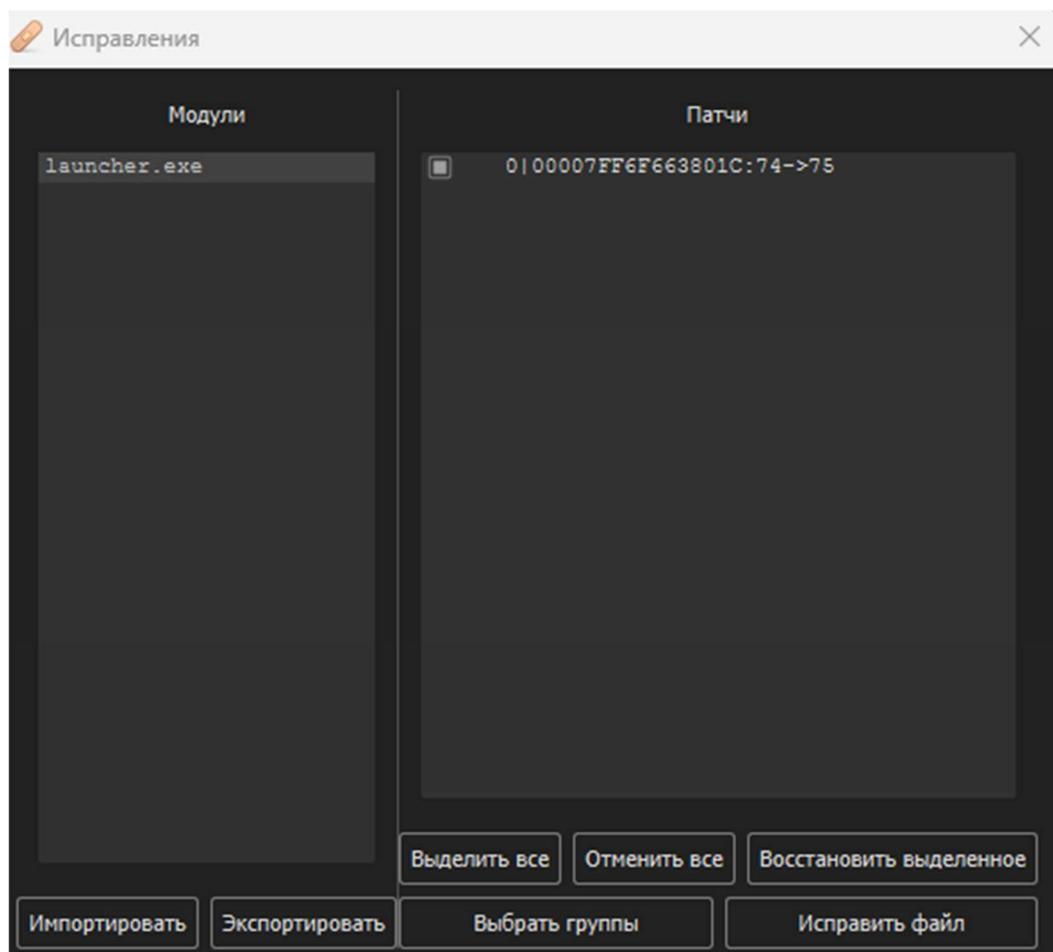
Чтобы не было перехода, нужно заменить је на jne

То есть опкод 74 на 75

Жмем Ctrl+E, меняем



## Сохраняем исправление



Сохраняем патч как отдельную программу launcher\_patched.exe, или заменяем оригинальный лаунчер на него

Запускаем MainProgramm

Вводим любой ключ, получаем флаг

```
Enter key: dasdasd
OK
Access granted. Here is the flag:
FECTF{b4ck_t0_zeroes}
|
```